UNITED STATES PATENT APPLICATION

for

DIGITAL THROTTLE FOR MULTIPLE OPERATING POINTS

Inventors:

James S. Burns
1245 Montclaire Way
Los Altos, CA 94024
Citizen of The United States

Stefan Rusu
1050 Eugene Court
Sunnyvale, CA 94087
Citizen of The United States

David J. Ayers
37884 Menard Court
Fremont, CA 94536-5816
Citizen of The United States

Edward T. Grochowski
5565 Yale Drive
San Jose, California 95118
Citizen of The United States

Marsha Eng
868 Rosemary Terrace
Sunnyvale, CA 94086
Citizen of The United States

Vivek Tiwari
373 River Oak Circle, Apt. 502
San Jose, CA 95134
Citizen of India

File No.: 042390.P12492

DIGITLE THROTTLE FOR MULTIPLE OPERATING POINTS

Related Patent Applications

This patent application is related to U.S. Patent Application No. _____, "Multiple Mode Digital Throttle Mechanism", filed on even date herewith and assigned to the same assignee.

Background of the Invention

Technical Field The present invention relates to microprocessors and, in particular, to mechanisms for controlling power consumption in microprocessors.

Background Art. Modern processors include extensive execution resources to support concurrent processing of multiple instructions. A processor typically includes one or more integer, floating point, branch, and memory execution units to implement integer, floating point, branch, and load/store instructions, respectively. Register files and on-chip caches are also provided to supply the execution resources with operands. When fully engaged by an executing program, these resources can create significant power dissipation problems.

Instruction code sequences that include enough instructions of the correct type to fully engage a processor's execution resources for significant intervals are relatively rare. Smart compilers and out of order execution can only extract so much instruction level parallelism (ILP) from most code. To conserve power, a processor may employ a clock gating mechanism to cut off the clock signal delivered to execution resources or their components that are not used by an executing code sequence. Such a processor can engage extensive resources as needed, e.g., to support code sequences with high ILP, without dissipating large amounts of power when code sequences with more typical ILP levels execute.

For code sequences having high ILP, few if any resources can be gated off, and the processor can dissipate significantly greater power than it does running code characterized by more typical ILP. To accommodate power-hungry code, a processor may be run at less than its top performance level by, for example, limiting its operating frequency. Hobbling the processor in this manner leaves a thermal margin for those code sequences that cause the processor to dissipate large amounts of power.

An alternative strategy, called power throttling, allows the processor to operate at its top performance level by default and reduces (throttles) the performance level if the processor's power consumption becomes too great. Power throttling may be implemented by, for example, temporarily reducing the number of instructions dispatched per clock cycle (instruction throughput), the frequency and/or the voltage at which the processor operates. This allows the processor to operate at its top performance levels for most code sequences. If a code sequence with extensive ILP runs, the processor reduces its performance level to maintain its power consumption within an established limit.

Regardless of how power consumption is controlled, effective power throttling requires a reliable mechanism for monitoring a processor's power consumption state. One currently available mechanism monitors the temperature of the processor's die to determine if throttling is required. An advantage of this approach is that it measures the consequence of excessive power consumption (die temperature) directly. A disadvantage is that changes in die temperature are slow relative to the time scale on which processors operate, e.g. on the order of seconds. This limits the speed with which high power consumption states can be detected and controlled. It also introduces analog circuitry into the predominantly digital environment of a processor, and if

the voltage or frequency of operation is altered to address power consumption, the monitor circuitry may be affected.

The present invention addresses these and other problems associated with monitoring and controlling power consumption by processor and other programmable devices.

Brief Description of the Drawings

The present invention may be understood with reference to the following drawings, in which like elements are indicated by like numbers. These drawings are provided to illustrate selected embodiments of the present invention and are not intended to limit the scope of the invention.

Fig. 1 is a block diagram of one embodiment of a computer system on which the present invention may be implemented.

Fig. 2 is a block diagram of one embodiment of a processor that implements a digital power throttle in accordance with the present invention.

Fig. 3 is a block diagram of one embodiment of a digital power throttle implemented by the processor of Fig. 2.

Fig. 4A is a block diagram of one embodiment of the monitor circuit of the digital power throttle of Fig. 3.

Fig. 4B is a block diagram of an alternative embodiment of the scalable current computation unit of Fig. 4A.

Fig. 5 is flowchart representing a method in accordance with the present invention for adjusting the power consumption of a processor.

Detailed Description of the Invention

The following discussion sets forth numerous specific details to provide a thorough understanding of the invention. However, those of ordinary skill in the art, having the benefit of this disclosure, will appreciate that the invention may be practiced without these specific details. In addition, various well-known methods, procedures, components, and circuits have not been described in detail in order to focus attention on the features of the present invention.

The present invention provides a mechanism for controlling the power dissipation of a processor under different operating regimes. An operating regime may be defined, for example, by the frequency of a clock signal employed by the processor, a voltage at which the processor's logic operates or a combination of frequency and voltage values.

In accordance with the present invention, a computer system includes a processor having a digital throttle. The digital throttle employs a weighted sum of activity states, scaled to reflect the current operating point of the processor, to estimate the power consumption of the processor. The digital throttle adjusts the processor's operating point if the estimated power consumption reaches a threshold value, and the scaling factor(s) is adjusted to reflect the new operating point.

For one embodiment of the invention, the digital throttle monitors the activity states of selected units of the processor's instruction pipeline in response to a sequence of instructions. The activity state of a unit indicates if the unit is gated on or off during a particular clock cycle. For each monitored unit, a weight representing, e.g., current or power consumed by the unit in its present activity state is provided. For example, a first weight may be provided if the unit is in a first activity state, and a second weight may be provided if the unit is in a second activity state. In effect, the digital throttle estimates power consumption of the processor through a weighted sum of activity states for the monitored units. This weighted sum is scaled for the current

processor operating point and tracked over a specified interval to provide an estimate of the processor's power consumption level (power state). If warranted by the estimated power state, the digital throttle signals a change in the processor's operating point, and a scaling factor, appropriate for the new operating point, is selected.

Scaling factors representing a range of processor operating points may be tabulated and stored for use by the digital throttle. A scaling factor suitable for the current operating point of the processor is applied to the weighted sum. Alternatively, weights for each monitored unit may be scaled, responsive to a change in the processor's operating point. In addition, a threshold value used to assess the power consumption of the processor may also be scaled, responsive to a change in the processor's operating point.

The scaling factor(s) may be estimated or determined through a calibration process. For example, the digital throttle may be calibrated once as a part of the design process or it may be self-calibrating. In the latter case, the digital throttle may employ current monitoring circuitry and a calibration algorithm periodically to adjust the scale factor(s) at different operating points.

The disclosed mechanism thus relies on digital events (activity states) in the processor's logic to estimate a power consumption level for the current operating point of the processor. The operating point may be adjusted, responsive to the estimate power state, and the scaling factor(s) applied to assess the power consumption is adjusted appropriately. This provides a fast, direct, and deterministic mechanism for controlling a processor's power consumption, and it does so without introducing analog circuitry into the processor.

Fig. 1 is a block diagram of one embodiment of a computer system 100 in which the present invention may be implemented. Computer system 100 includes one or more processors 110, a main memory 140, a non-volatile memory 150, various peripheral devices 160, and

system logic 170. System logic 170 controls data transfers among processor(s) 110, main memory 140, non-volatile memory 150, and peripheral devices 160. A power delivery unit 180 is also shown. Computer system 100 is provided to illustrate various features of the present invention. The particular configuration shown is not necessary to implement the present invention.

Processor 110 includes multiple units 124, which form an instruction execution pipeline 120. Instructions are provided to processor 110 from main memory 140 and non-volatile memory 150. A digital throttle 130 monitors power consumption in the various units 124 in response to the processed instructions and adjusts operation of processor 110 accordingly.

As an instruction is staged down pipeline 120, it directs various units 124 to perform one or more operations that, taken together, implement the instruction. For example, a floating-point multiply-accumulate instruction (FMAC) may cause the following operations to occur in the indicated resources: a floating point register file reads out three operands; an FMAC execution unit multiplies two of the operands and adds the product to the third operation; an exception unit checks the product and sum for errors; and a retirement unit writes the result to the floating point register file if no errors are detected. Depending on the particular processor implementation, these resources or their components may be grouped into one or more units which are turned on and off as the instruction is staged down the pipeline. Each unit consumes a certain amount of current as it is activated by the instruction.

For one embodiment of the present invention, the current or power consumed by a unit 124 is represented by an associated weight or weights. When a unit is activated by an instruction, digital throttle 130 detects its active state and includes a first weight in a sum representing the activity, e.g. current or power consumption, of the processor on a given clock

cycle. If the unit is not activated, digital throttle 130 may add a second weight to the sum. The second weight may be zero or a value representing current or power consumption of the unit in the inactive state. For one embodiment, the resulting weighted sum is scaled to reflect the processor's current operating point. For another embodiment, the individual weights may themselves be scaled. Digital throttle 130 implements these operations over a selected interval, generates a value representing an estimated power state for the processor, and adjusts the processor's operation if the value exceeds a specified threshold level.

Fig. 2 represents in greater detail one embodiment of processor 110. For the disclosed embodiment of processor 110, pipeline 120 is represented as fetch (FET), expand (EXP), register (REG), execution (EXE), detect (DET), and retirement (RET) stages, respectively, and the execution resources corresponding to each stage are indicated. The present invention does not require partition of processor 110 into a particular set of pipeline stages. For example, a disclosed stage may be subdivided into two or more stages to address timing issues or facilitate higher processor clock speeds. Alternatively, two or more stages may be combined into a single stage. Other embodiments may include hardware for processing instructions out-of-order. The disclosed pipeline provides only one example of how operations may be partitioned in a processor implementing the present invention.

The front end of pipeline 120 includes fetch unit 210 and issue unit 220, which provide instructions to execution units in the back end of pipeline 120 for execution. Fetch unit 210 retrieves instructions from memory 140 directly or through a local cache (not shown) and provides the fetched instructions to issue unit 220. Issue unit 220 decodes the instructions and issues them to the execution resources in the back end of pipeline 120.

Throughout this discussion, the term "instruction" is used generally to refer to instructions, macro-instructions, instruction bundles or any of a number of other mechanisms used to encode processor operations. For example, the decode operation may transform a macro-instruction into one or more micro-operations (μops), resolve an instruction bundle into one or more instruction syllables, or retrieve a micro-code sequence associated with an instruction.

The back end of pipeline 120 includes register unit 230, execution unit 250, exception unit 260 and retirement unit 270. Register unit 230 includes a register rename unit and various register files (not shown) to identify the registers specified in the instructions and to accesses the data from the identified registers, respectively. Execution unit 250 includes one or more branch execution units (BRU) 252, integer execution units (IEU) 254, load/store units (LSU) 256, and floating point execution units (FPU) 258 to process branch, integer, load/store, and floating point instructions. Exception unit 260 checks the results generated by execution units 250 and adjusts the control flow if an exceptional condition is encountered. If no exceptional conditions are detected, retirement unit 270 updates the architectural state of processor 110 with the results.

The units activated by different instructions correspond to various combinations and subsets of the execution resources indicated for pipeline 120. For example, one unit may include a floating-point register (in register unit 230), and FPU 258 may have components in two or more units. In general, a unit includes various execution resources (register files, execution units, tracking logic) that are activated and deactivated together. The present invention does not depend on the detailed mapping between the units and the execution resources shown in Fig. 2.

Fig. 3 is a block diagram representing one embodiment of digital throttle 130 and its interactions with units 124 of pipeline 120. The disclosed embodiment of digital throttle 130 includes gate units 310(1)-310(n) (generically, gate unit 130), a monitor circuit 320, and a

throttle circuit 350. Each gate unit 310 is associated with a unit 124 in pipeline 120 to monitor and/or control current delivery to the unit. For example, gate unit 310 may be a clock gating circuit that couples or decouples a clock signal to unit 124 according to whether or not the services of unit 124 are necessary to implement an instruction currently in the pipe stage in which the unit operates. Monitor circuit 320 collects signals from gate units 130 and assesses the current power state for processor 110 from the collected signals. Also shown in Fig. 3 is a pipeline control circuit 314, which indicates to gate units 310 which units are active for the currently executing instructions.

For the disclosed embodiment of digital throttle 130, each gate unit 310 provides a signal to monitor circuit 320 to indicate an activity state of its associated unit 124. For example, the signal may be asserted if unit 124 is turned "on", in which case, monitor circuit 320 adds a first weight to an estimate representing the per clock activity of processor 110. The first weight may represent the current or power provided to unit 124 when it is activated. For this embodiment, if unit 124 signal is not "on", the signal is deasserted, and monitor circuit 320 adds a second weight to the estimate. The second weight may represent the current or power provided to unit 124 when it is deactivated. The second weight may be zero, a value representing a leakage current in deactivated unit 124 or a similar value representative of its deactive state. A typical processor may include 10-20 gate units 310 to control power delivery to 10-20 units 124.

For the disclosed embodiment of digital throttle 130, monitor circuit 320 includes an activity monitor (AM) 330, a conversion unit (CU) 340, and a threshold unit (TU) 350. Also shown is a throttle unit 360 which adjust the operating point of a processor including digital throttle 130 if indicated by threshold unit 350.

AM 330 provides a clock by clock estimate of the processor's overall activity, based on the activity states of monitored units 124 and the current operating point of the processor. CU 340 accumulates per clock estimates and provides the accumulated value to TU 350 for comparison with a power threshold. For one embodiment of the invention, CU 340 may also be adjusted, responsive to a change in the operating point of the processor. TU 350 determines if the accumulated power value reaches a threshold value and signals a throttle unit to adjust the processor operation appropriately. For example, if the threshold value indicates a high power consumption state, throttle unit 360 may trigger the processor to enter a lower power consuming state. Lower power consumption states may be reached by reducing the instruction issue rate, reducing the frequency at which the processor operates, reducing the voltage at which the processor's logic operates or combinations of these and other power reduction mechanisms.

The power/current consumed by the processor's units depend the operating point of the processor. For example, at lower frequencies and/or voltages, a given unit consumes less power in its activated state than it does at higher frequencies and/or voltages. Accordingly, AM 330 adjusts its activity estimation to reflect changes in the operating state. For other embodiments of digital throttle 130, CU 340 may also adjust its computations to reflect changes in the processors operating state.

Fig. 4A shows in greater detail one embodiment of monitor circuit 320. For the disclosed embodiment of monitor circuit 320, AM 330 includes weight units 432(1) - 432(n) (generically, weight units 432), an adder 434, a first scale unit 436, and a storage unit (SU) 438 for scale factors. Each weight unit 432 is associated with one of units 124 through a corresponding gate unit 310. Weight unit 432 provides a first weight to adder 434 if the activity state signal from its gate unit 310 is asserted, and a second weight if the activity state signal is not asserted.

For units that are not clock gated, AM 330 may include a weight representing the relative power or current these always-active units consume.

Adder 434 sums the weights indicated by weight units 134 and first scale unit 436 scales the sum to reflect the current operating point of the processor. For the disclosed embodiment of digital throttle 130, SU 438 stores scale factors appropriate for different operating points and provides the appropriate factor to scale unit 436. SU 438 may be implemented, for example, as a look-up table, cache or comparable data storage element. If an operating point input (OP1) indicates a change, SU 438 provides a new scale factor to scale unit 436.

For the disclosed embodiment of monitor circuit 320, CU 340 includes an adder 442, threshold storage unit 444, and an accumulator 446. A value in storage unit 444 may represent, for example, a current or power threshold. The threshold value is negated and provided to an input of adder 444, the second input of which is coupled to the output of scale unit 435. The difference is added to a value in accumulator 446, which effectively integrates the per cycle activity relative to the threshold value. For example, the overall activity may be scaled represent an estimate of the per cycle power consumption of the processor and the first threshold value may be a specified average power. The value in accumulator 446 may then represent a thermal excess or deficit, relative to a temperature associated with the average power state.

Other embodiments of CU 340 may include circuitry to forward the value in accumulator 446 to threshold unit 350 at a specified sampling frequency. The sampling period allows digital throttle 130 to moderate the impact of power consumption spikes. Other embodiments of digital throttle 130 may use the output of adder 444 to control throttle unit 360 directly, e.g. on a clock by clock basis.

Also shown in Fig. 4A is a second, optional scaling unit 448 to apply a selected scale factor to the threshold value stored in threshold storage unit 444. Scaling unit 448 allows the value in threshold storage unit 444 to be scaled, responsive to changes in the processor's operating point, indicated by OP2. Persons skilled in the art will recognize that adjustment of the threshold value may be implemented in other ways as well. For example, another embodiment of CU 340 may store multiple threshold values and select a threshold value responsive to the operating point data indicated by OP2.

For the disclosed embodiment of monitor circuit 320, the value tracked by accumulator 446 (accumulated power consumption) is provided to TU 350 for comparison with a second threshold value. The disclosed embodiment of TU 350 includes a storage 454 and a comparator 458. As noted above, the value in accumulator 446 may be provided on each clock cycle or following a specified sampling period. Storage 454 stores a value representing the second threshold level against which the accumulated power consumption is compared. For the example described above, the second threshold may represent upper and lower temperature thresholds that trigger power reducing and power increasing mechanisms, respectively, if reached.

If the accumulated activity reaches the second threshold, TU 350 provides a signal to throttle unit 360, which triggers an appropriate adjustment to the power consumption level of processor 110. As noted above, the adjustment may be accomplished by issuing fewer (more) instructions per clock cycle, reducing (increasing) the clock frequency at which the processor operates, reducing (increasing) the voltage at which it operates, or some combination of these strategies or other power reduction (increasing) strategies. To the extent any of these changes affect the weighting appropriate for monitored units 124, new scale factor(s) are selected for use by digital throttle 130.

Fig. 4B is a block diagram showing another embodiment of AM 330 (AM 330'). For the disclosed embodiment, SU 438 adjusts the values of weight units 432, responsive to a change in the operating point of the processor.

Fig. 5 is a flowchart representing a method 500 in accordance with the present invention for controlling power consumption in a processor. Method 500 estimates 510 the activity of the processor, based on active/inactive states of its component units. The estimated activity is then scaled 520 according to the current operating point of the processor to provide an indication of the processor's power state. The scaled activity is compared 530 with a threshold. If the comparison 530 indicates the processor is operating outside a specified regime, the processor's operating point is adjusted 540 and a scale factor appropriate for the new operating point is selected 550. If the comparison indicates the processor is operating within the specified regime, method 500 continues with the current scale factors. For embodiments of method 500, the scaled activity may be tracked or integrated over time prior to comparison, to mitigate the impact of power spikes.

As noted above, different mechanisms may be employed to adjust the power state of a processor, responsive to an indication from the processor's digital throttle that the processor is operating outside a specified regime. For example, power consumption changes in a substantially linear manner with changes in the processor's instruction throughput. One mechanism for adjusting instruction throughput introduces "bubbles", e.g., empty instruction slots, in the normal flow of instructions or otherwise reduces the number of instructions dispatched per clock cycle. This may be accomplished by injecting No-Operations (NOPs) into the instruction flow at a particular rate. The higher the ratio of No-Operations to useful, e.g. power-consuming, instructions, the lower the power consumption state of the processor is. U.S.

Patent Application No.09/471,795 describes mechanisms for adjusting instruction throughput using a digital throttle.

Linear power reduction mechanisms, such as reducing instruction throughput, can be implemented with relatively low latencies, e.g. on the order of a couple of clock cycles. However, their ability to reduce power may not be sufficient to address a power state detected by digital throttle 130. Adjusting the processor's operating frequency, its operating voltage, or some combination of the frequency and operating voltage has a greater impact on power consumption. The operating point of a processor may be altered by, for example, reducing the clock frequency (ν) and/or the voltage (V) at which the processor operates. The frequency at which a processor can operate scales linearly with V, due to the lower transistor gate drive available at lower voltages, and power dissipation scales with the square of the voltage (V^2). Reducing the frequency and voltage together thus reduces the power consumption at a rate that scales as V^3. Power throttling by scaling ν and V can be very effective, but changing the operating point of the processor is a relatively slow process.

A digital throttle may be used to adjust instruction throughput first, if a monitored activity indicates the approach of a high power state. Since this power reduction strategy does not alter the current or power estimates based on processor activity, no adjustment of the scale factor is necessary. If a reduction in instruction throughput does not reduce power consumption significantly, or if throughput reduction is triggered too frequently, the digital throttle switches to voltage and frequency-based mechanisms to control power consumption. Voltage-based strategies alter the processor's operating point, which in turn alters the mapping between activity states and power consumption. The scale factor(s) is adjusted responsive to a change in the operating point.

There has thus been provided a digital throttle that monitors the activity of a processor and scales the monitored activity according to the processor's current operating point to estimate its power consumption. If the estimated power consumption falls outside a specified range, the processors operating point is adjusted, a scaling appropriate for the new operating point is selected, and monitoring proceeds at the new operating point.

The disclosed embodiments have been provided to illustrate various features of the present invention. Persons skilled in the art of processor design, having the benefit of this disclosure, will recognize variations and modifications of the disclosed embodiments, which none the less fall within the spirit and scope of the appended claims.

We claim: